# *Using High Level Architecture for Electronic Warfare Test and Evaluation*

*A Technical Paper*
*from the*
*Joint*
***A**dvanced*
***D**istributed*
***S**imulation*
*Joint Test Force*

*Mr. Clyde Harris, Science Applications International Corporation*

Presented at the Royal Aeronautical Society Symposium
*Networking in Simulation and Training -- Realising the Benefits*
11-12 November 1998, London, UK

JADS JTF
http://www.jads.abq.com
11104 Menaul NE
Albuquerque, NM 87112-2454
(505) 846-1291
FAX (505) 846-0603

# USING HIGH LEVEL ARCHITECTURE FOR ELECTRONIC WARFARE TEST AND EVALUATION

Mr. Clyde Harris, Science Applications International Corporation
Joint Advanced Distributed Simulation Joint Test Force
Albuquerque, New Mexico USA

**<u>Abstract</u>**

The Joint Advanced Distributed Simulation (JADS) Joint Test Force (JTF) was chartered by the Office of the Secretary of Defense (Acquisition and Technology) to investigate the utility of Advanced Distributed Simulation (ADS) technology, including the High Level Architecture (HLA), for use in Test and Evaluation (T&E). To accomplish this, the High Level Architecture (HLA) and a Run Time Infrastructure (RTI) are being used by JADS to perform test and evaluation of an electronic warfare self protection jammer (SPJ).

The Electronic Warfare (EW) test is collecting SPJ performance data in three different phases of system development and test. The three JADS EW test phases are: a linked test of an EW Digital System Model, a linked test of a SPJ in an Installed System Test Facility, and an Open Air Range flight test. The linked tests will employ an ADS test architecture, including the HLA, for evaluating representations of the EW system under test (SUT) early in the system design phase and late in the development phase. For the linked tests, JADS will use a distributed network of facilities across the United States. The emphasis of the JADS EW test is on the performance of the ADS components and their benefit, if any, to EW T&E, rather than on any particular EW system under test or class of systems. The JADS EW linked tests are designed to evaluate issues associated with distributed testing, network performance, and their relationship to EW performance data collected.

This paper provides JADS experience to date in applying Defense Modeling and Simulation Organization's (DMSO) HLA in the JADS EW test. The process used by JADS to design the EW test, characterize network latencies, the RTI performance measurements made in the JADS test bed, and how that characterization will be applied to the JADS federation are provided.

## <u>Overview</u>

The JADS JTF is Air Force led, with Army and Navy participation, and is scheduled for completion in 1999. The JADS EW test program adopted the HLA to formally evaluate its capabilities and gather lessons learned for the DoD T&E community. The JADS implementation of HLA methodology included building the federation object model (FOM) to support JADS test requirements, defining RTI performance requirements for the JADS EW test, developing a test methodology and tools to characterize network and RTI latency, taking measurements in a JADS test bed, and applying results to the JADS federation and distributed test architecture.

## <u>Test Approach</u>

The JADS EW test focus is on evaluating the utility of Advanced Distributed Simulation. The test applies the HLA processes and RTI to gather some initial insights into their use in an actual EW test of a Self Protection Jammer (SPJ). The HLA is an integral component of JADS EW test design, and represents a key element of the building block approach JADS will use to determine the current capability ADS provides for EW T&E. There are significant technical challenges to fully implementing ADS in EW T&E that must be evaluated. JADS EW applies a building block approach to determining the extent to which ADS may be used in EW T&E by comparing EW measures of performance (MOP) from flight test with MOP data collected using an ADS test architecture. The issues and objectives developed for the JADS EW test are listed below.

**Table 1 - JADS EW Issues and Objectives**

| Issues | Objectives |
|---|---|
| Issue 1: What is the present utility of ADS, including DIS, for T&E ? | Objective 1-1: Assess the validity of data from tests using ADS during test execution.<br>Objective 1-2: Assess the benefits of using ADS in T&E |
| Issue 2: What are the critical constraints, concerns, and methodologies when using ADS for T&E ? | Objective 2-1: Assess the critical constraints and concerns in ADS performance for T&E.<br>Objective 2-2: Assess the critical constraints and concerns in ADS support systems for T&E.<br>Objective 2-3: Develop and assess methodologies associated ADS for T&E. |
| Issue 3: What are the requirements that must be introduced into ADS systems if they are to support a more complete T&E capability in the future ? | Objective 3-1: Identify requirements for ADS systems that would provide a more complete T&E capability in the future. |

## EW SPJ Test Scenario

The test scenario consists of a single penetrating aircraft crossing into opposition airspace on a strike mission. The opposition target is protected by an integrated air defense system. The penetrating aircraft will engage three surface-to-air missile batteries and one anti-aircraft artillery system during ingress and egress. The figure below illustrates the JADS EW scenario.

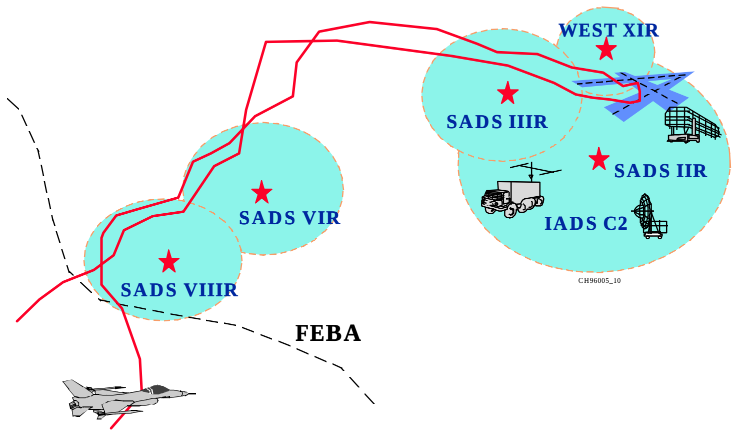


**Figure 1. SPJ Test Scenario**

## Self-protection Jammer Test Phases

The three-phases of the EW test focus on different developmental stages of a SPJ and the EW test process.

Phase I of the test will consist of a series of runs on an OAR against the threat laydown. The aircraft will fly a scripted profile across the simulated battle area (FEBA) and have engagements with the various simulated threats. The aircraft will fly a single flight profile to collect a population of samples of performance data for a single reference test condition. The purpose of this test is to establish a baseline of performance data which will be used to develop the ADS test environment for the following test phases and will be the basis for determining the validity of ADS test results. Although flight testing of the SPJ would normally occur at the end of the EW test process, JADS is conducting this phase first to collect the data required to replicate the test in the ADS environment. Results from the ADS test phases will be compared to the results from the Phase I OAR test to assess if an ADS test environment (like Phase II and III) will produce data results similar to the OAR test. Phase II and III of the EW test use ADS for the test architecture.

The Phase II test diagram below identifies the facilities and the types of data interchanges required to duplicate the Phase I test scenario.

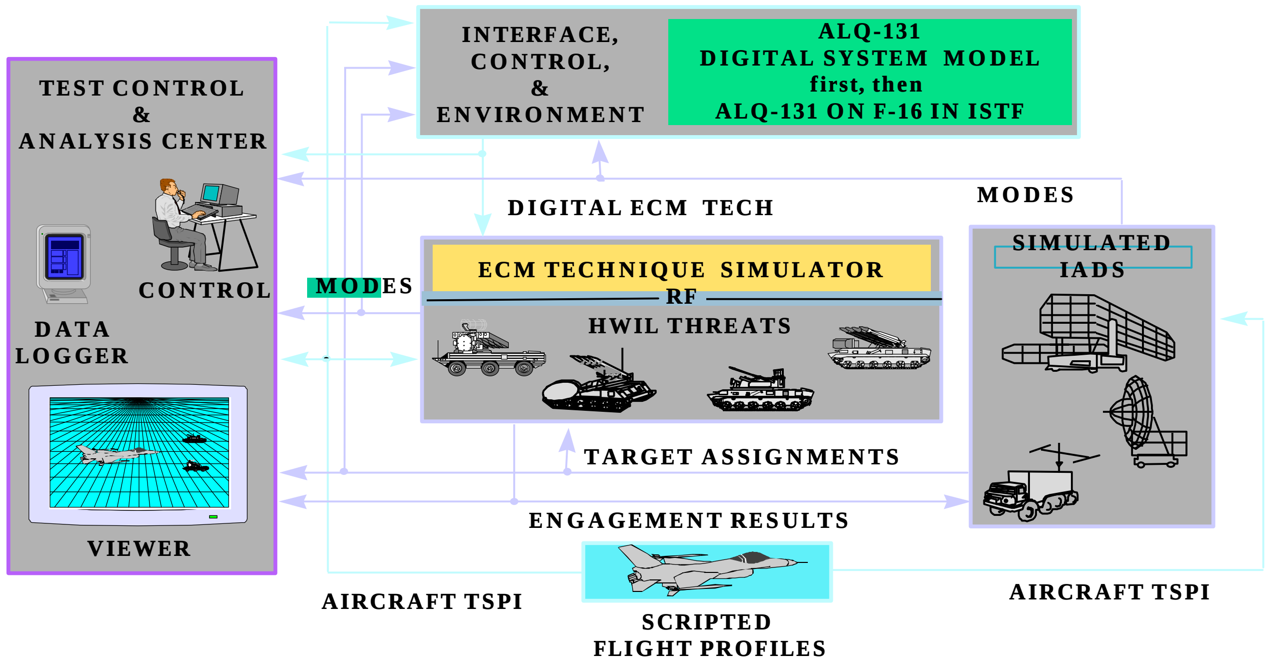


**Figure 2. JADS EW Test Using an ADS Architecture**

Phase II is an ADS based EW test using a high-fidelity real-time digital system model (DSM) of the SPJ, models, and simulations. The threat laydown from the OAR will be replicated in the ADS environment including links with man-in-the-loop terminal threat simulations. The SUT will be flown, via a scripted flight profile developed from the actual OAR flights, to a designated target protected by high-fidelity terminal threats controlled by an air defense system. The DSM would normally be developed as a tool for requirements analysis early in the development of a new EW system and would not typically be tested in a high-fidelity threat environment. This test will evaluate the ability to apply increased threat fidelity and resources using ADS test capabilities much earlier in the development cycle.

Phase III is also an ADS test using the SPJ installed on an actual aircraft located in an Installed System Test Facility (ISTF). The SUT in the ISTF will interact with the simulated threats and the air defense system using the same threat laydown as the previous tests and controlled by the same scripted flight profile. In the normal EW process, ISTF testing is used late in the development cycle to measure the effect of aircraft systems on the performance of the SPJ.

A key requirement for the test designer using ADS is to tailor the distributed test environment based on the maturity of the system, the test objectives, and program constraints. JADS have applied a similar approach to the selection of components for this test.

EW SUT : The ALQ-131 Block II with reprogrammable processor is the SUT. Key to its selection was the availability of a DSM. During the development of the ALQ-131 Block II R/P, a digital model was developed which simulates appropriate functions of the SPJ in the JADS Phase 2 test.

Aircraft platform : Along with compatibility with the SUT, we required the selected host aircraft to be a non-developmental penetrator. The aircraft needed to be readily available and capable of being easily supported and instrumented for the tests. The logical choice was the F-16C.
EW Terminal Threats : The primary facility in any of the military services for simulating terminal threats to aircraft is the Air Force Electronic Warfare Environment Simulator (AFEWES) facility at Ft Worth, TX. The facility provides a wide variety of terminal threat simulators and has considerable linking experience.

## **JADS EW Federation**

Phase II uses software models and man-in-the-loop simulators linked together using the HLA to replicate key elements of the Phase 1 test. Test data similar to the data collected from the Open Air Range (OAR) test will be generated in each of the ADS test phases to compute the same set of MOPs. The models and simulators (in HLA, referred to as federates) are linked together via the RTI and comprise a HLA federation. A federation is a named set of interacting federates, with a common federate object model (FOM) and supporting RTI, that are used as a whole to achieve some specific objective.

Each OAR run is reduced into time-ordered “data scripts” representing the time, space, position information (TSPI) of the aircraft, command and control (C2) threat handoff commands, and the background RF environment. The interactions between the AFEWES threats and the DSM will be dynamic based upon aircraft location, the AFEWES man-in-the-loop operations, and capabilities of the SUT against a specific simulated threat. Test execution and control will be handled using computers at JADS to start and stop tests and for status monitoring. The JADS EW federation consists of six federates described below and shown in Figure 3.

- Test Control Facility (TCF) - The TCF federate manages the test execution, collection of necessary raw data to evaluate SPJ performance and to reduce raw data into MOPs.

- Terminal Threat Handoff (TTH) - The TTH federate is responsible for assigning terminal threats at the THREATS federate to the target simulated by the PLATFORM federate. Initial position data is provided by this federate to the THREATS for target acquisition.

- Radio Frequency Environment (RFENV) - This federate is responsible for reporting extraneous emissions detected during the OAR tests to the DSM federate. In order to replicate the test environment from the OAR test phase during the ADS phases, these extraneous emissions must also be simulated in the ADS test environment for injection into the SUT.

- Digital System Model (DSM) - This federate is responsible for providing the simulation of the AN/ALQ-131 jammer and its RF environment. During the Phase II ADS test, this simulation of the SUT consists of an all-software simulation of the jammer as well as the RF environment presented to it.

- Aircraft PLATFORM - This federate serves two purposes within the JADS federation. Its primary function is to provide the positional information for the test aircraft to the other federates. This aircraft position is based on GPS-recorded data from the OAR test missions. The second function of the federate is to report target tracked position from the OAR test missions.

- THREATS - This federate is responsible for providing the terminal threat simulations. These simulations include human-operated threat simulations designed to track a simulated target. THREATS will also provide an RF simulation of the SPJ technique waveforms.

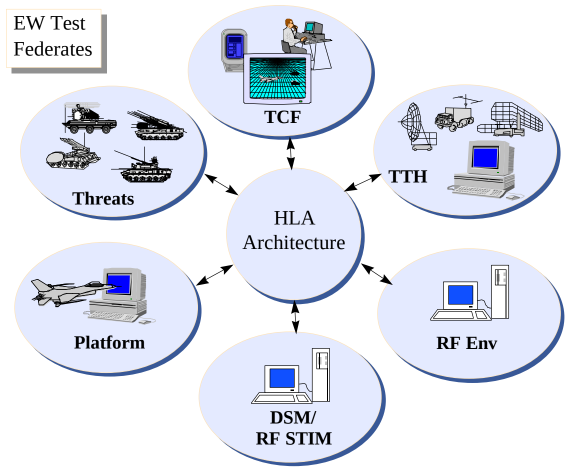


**Figure 3. JADS EW Federates**

The only difference between Phase 2 and Phase 3 ADS architectures is the representation of the SUT. The test scenario and all other federates remain the same.

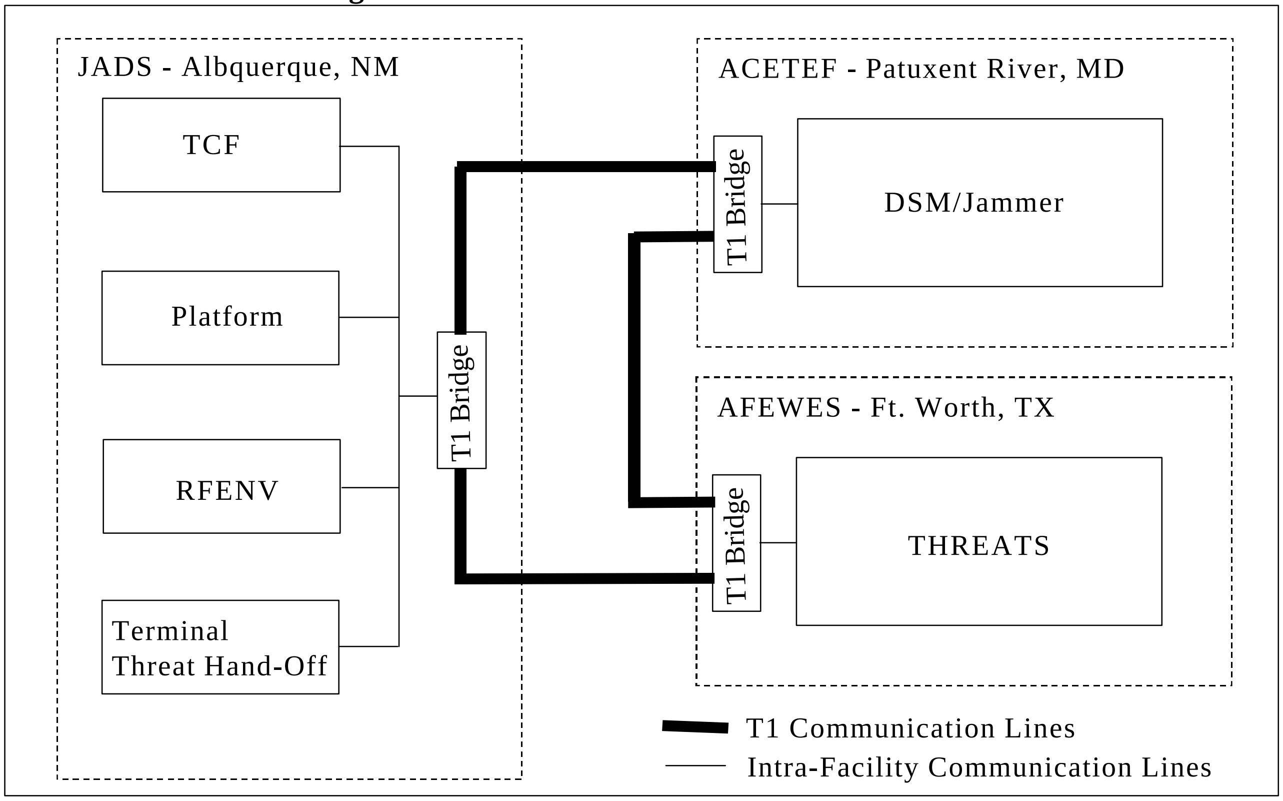
## Federation Configuration

The primary tool for documenting and communicating detailed JADS requirements to DMSO and the RTI development community was the Federation Execution Planners Workbook [2]. The workbook contains extensive descriptions of the JADS federates, attributes and interactions, computers and communication equipment and RTI services required.  To summarize, the JADS EW federation uses dedicated T-1 circuits, communications, and encryption devices to link JADS with its two key EW test facilities, AFEWES and ACETEF.  Three network nodes interconnect the six JADS federates used in Phase II.  Four of the six federates execute on dedicated Silicon Graphics, Inc. (SGI) O2 workstations in the JADS test control facility at Albuquerque, New Mexico.  The DSM federate executes on an SGI O2 at Air Combat Environment Test and Evaluation Facility (ACETEF) and the THREATS federate executes on an SGI Challenge computer at the AFEWES facility.  The federates at JADS will publish attributes at a combined rate of 20 Hertz (Hz).  The worst case instance of the AFEWES federate will have 11 attributes published at 20 Hz.  The ACETEF federate will publish 1 attribute at 20 Hz.  All nodes will publish interactions at approximately 1 Hz.  The largest JADS federation attribute or interaction is 106 bytes in length.  The network architecture and federate geographic locations are illustrated in figure 4.  The following table is a summary of the data interchange requirements for Phase II and III tests.

**Table 2.  JADS Federation Requirements**

| Performance Measure | JADS Requirement |
|---|---|
| Attribute/Interaction Size | Max: 672 bits Min: 16 Bits |
| Update Frequency | Max: 20 Hz Min: 1 Hz |
| Expected Bandwidth | Max: 183335 bits per second |
| Time to Create New Objects | 10 ms |
| Central Processing Unit (CPU) Utilization | RTI: 25% Overhead: 5% |
| Allowable RTI Latency | < 140 ms for closed-loop interaction |

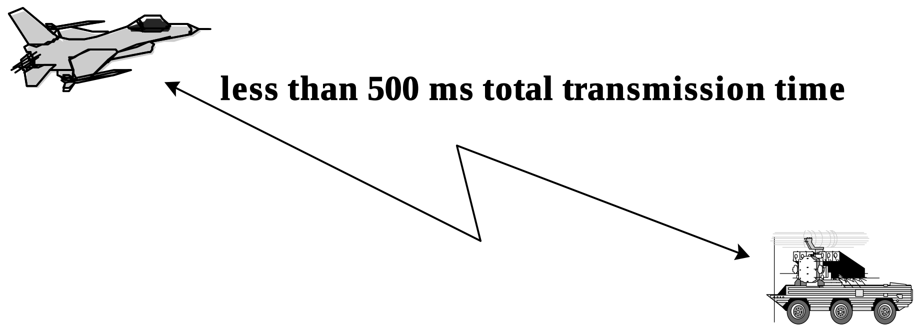
**Figure 4.  JADS EW Phase 2 Test Architecture**



## RTI Requirements and Test Methodology

In developing and implementing an HLA federation for EW T&E, JADS recognized that measuring and controlling the latency imposed by diverse test facilities, simulators, communications equipment, and long-haul communications networks was a critical factor. Because of the importance to T&E, most of these latency measurements have been documented by other EW test projects. The new element used by the JADS test for EW T&E is the HLA and accompanying RTI software. Since the RTI provides a new means for dissimilar simulators and facilities to communicate, an additional source of latency is imposed on a distributed test architecture which must be measured, optimized, and controlled for accurate real-time data collection and measurement of test events. The first step in the effort for the JADS EW test was to define the RTI performance requirements for the Phase 2 and Phase 3 tests.

The RTI performance requirements for JADS came from a solid understanding of the interactions between aircraft carrying self-protection jammers and surface-to-air threat systems in an OAR test environment. The problem space was defined by the reference test condition (RTC) used in the Phase I test. Closed-loop testing using ADS technology runs the risk that the communications infrastructure used to carry out the interactions among objects will change the outcome either through lost interactions, or by changing the temporal nature of the interaction. This temporal change is usually an increase in the time for the interaction to occur called latency. The amount of allowable latency depends on the nature of the interactions and the decision cycle of each system. The EW test interaction of interest is the threat radar activation, jammer identification and response, and associated threat response. Depending on how the engagement is carried out, the interaction can be the jammer's computer working against the threat's computer or the jammer's computer working against the threat's human operator. Due to physical limitations in distributed testing, JADS chose to look at the latter interaction exclusively. Therefore, the limitations that JADS placed on the communication infrastructure latency with human operator interaction is 500 milliseconds (ms). That means from the time the radar changes state, the infrastructure has 250 milliseconds to get that message to the jammer and another 250 milliseconds to return the jammer's response. We refer to this as an end-to-end interaction shown below.



Of the 250 milliseconds, the RTI is allocated 70 milliseconds. In the ADS environment, the ADS architecture will add additional latencies to the real latencies described above.

For the interaction shown above, the following steps occur in the ADS environment JADS will use :

1. Radar on
2. Radar state passes to the Air Force Electronic Warfare Evaluation Simulator (AFEWES) application program interface (API). The API is a library of function calls which allows a federation to interact with the RTI.
3. AFEWES API passes radar state to ACETEF API using reliable effort
4. ACETEF API passes radar state to Advanced Tactical Electronic Warfare Environment Simulator (ATEWES) to radiate radar radio frequency (RF).
5. Jammer initiates a response
6. Jammer instrumentation captures response and transmits to ACETEF API
7. ACETEF API passes jammer state to AFEWES API using reliable effort
8. AFEWES API passes jammer state to the JammEr Technique Simulator (JETS) to initiate RF
9. Radar receives jammer response

Steps 2, 3, 4, 6, 7and 8 introduce additional latency to the real-world interaction. Steps 3 and 7 are latencies introduced specifically by the RTI software and the geographical latency due to separation of the facilities. The expected JADS EW latency estimates which are the non-RTI latencies are

| | |
|---|---|
| Step 2 - | 50 ms |
| Step 4 - | 100 ms |
| Step 6 - | 60 ms |
| Step 8 - | 50 ms |
| Total - | 260 ms |

For reliable data transfer, JADS assumed that there will be one transfer from the publishing federate to the RTI's federation executive (FEDEX) and one transfer from the FEDEX to the destination or subscribing federate for both steps 3 and 7. The JADS facility located in Albuquerque will be the RTI FEDEX host. This adds 4 times the expected geographical latency for both RTI latencies (i.e., two geographical latencies per RTI transfer). Based on prior data, the geographical latency was measured as 25 ms (one way) between ACETEF and AFEWES.. The total non-RTI latency is therefore 260 ms + 4* 25 ms = 360 ms.

The maximum allowable latency is driven by the time necessary to initiate jamming when a radar is activated and the time necessary to terminate jamming when a radar beam is pulled off the target. The most critical time factor for initiating jamming is if the technique is designed to deny acquisition by the threat. In this instance the jamming must be presented to the radar within 500 ms. This value is based on the human response time (200 ms for visual recognition + 300 ms for physical reaction) to the technique. In the instance when the radar beam is pulled off the target, the jamming must terminate before the operator can reacquire the jamming signal. This time is again based on the human response time of 500 ms.

Based on these requirements, the maximum value of the RTI latency in steps 3 and 7 must therefore be the difference between 500 ms and 360 ms, or 140 ms total.

The maximum one way RTI latency is therefore 70 ms. The RTI latency is defined as follows:

Step 1. $API_{in}$ to network
Step 2. Network to RTI FEDEX
Step 3. RTI FEDEX to network
Step 4. Network to $API_{out}$

All network latencies between steps 1 and 2 and steps 3 and 4 have been included in the geographical latencies described above.

## RTI Testing

To determine RTI latency within the JADS EW test architecture, a test bed was established which included all the computer and communications components of the Phase II ADS architecture. With a test bed in place, specific test software tools were developed to exercise, log, and measure latency of the network with and without the RTI. There are two types of software JADS developed for the RTI tests. First, we developed software to send data one way between two computers. There are versions of this software that perform tests without an RTI (using both TCP and IP multicast), and there are versions that perform tests using the RTI (using reliable and best effort). RTI reliable traffic is sent using TCP transmission. RTI best effort traffic is sent using IP Multicast transmission. Using the JADS test bed configuration, the purpose of the non-RTI tests is to characterize the network performance in the simplest test case. The one-way software is designed to exercise the network and the RTI with different data sizes and transmit rates. The size is varied between 17, 51, 101, 301, 501, and 1001 bytes. The transmit rate is varied between 5, 10, 20, 50, 100, 200, 400, and 500 Hz. The complete matrix of rate and size combinations was tested. Each test case, defined by a specific rate and size pair, ran for thirty seconds. For the RTI version of the one-way software, a separate matrix was generated for attributes sent as reliable and best effort. Initially, only one test of each pair was run.

In all JADS RTI tests, latency and lost data are the two metrics examined. To track lost data, all of the messages (either attributes or interactions) contain a serial number. To calculate total latency, the "send" time is included in the message and when a message arrives, the "receive" time is saved. For this design to work, the system time for all the computers that participate in a test must be synchronized. In the JADS test federation, we use Datum BanComm GPS cards to accept an IRIG B or GPS input to synchronize time. However, these cards were not available when we began RTI testing so we used the Network Time Protocol (xntp) software to synchronize the time on all test computers. A GPS receiver provides time via its serial port to the Stratum-1 time server. All of the other computers in the network receive their time from this time server. It takes a few days to get the whole system initially configured and settled down. After that, the system time on all computers remained within 1 ms of GPS time. The xntp software generates statistics on how well it is keeping time. The Datum BanComm card was used to verify that the offset reported by the xntp software was accurate.

The RTI test hardware configurations progressively increased in complexity until the entire JADS federation and network architecture (except for the T1 lines) is in place in the test bed. Starting with a two computer point-to-point configuration, we gathered basic performance data for network IP Multicast data, network TCP data, and RTI reliable data. The two node test configuration used initially for all tests as shown in figure 5. All sources of possible latency were computed through a disciplined process of adjusting one variable at a time and collecting recorded time data for the same message type in differing reference test conditions. The two-node network test used an SGI O2 5000 and an SGI O2 10000 running IRIX 6.3. The test software and RTI version 1.3 were hosted on each computer for all tests using this configuration.

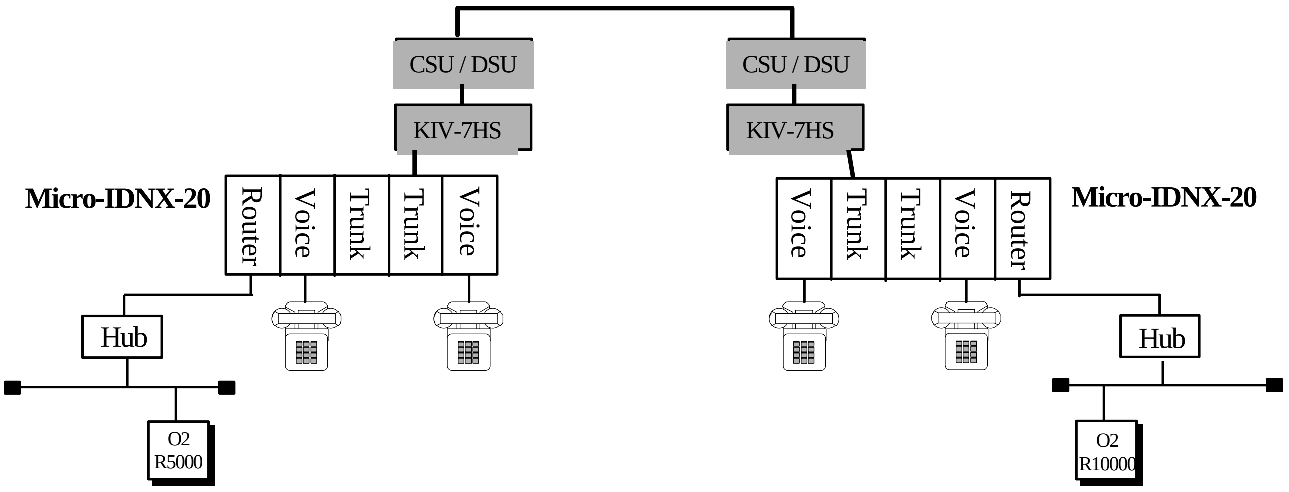


**Figure 5. JADS 2 Node Test Bed Configuration**

For the network IP Multicast tests (no RTI), the minimum, maximum, and mean latencies were between 7 and 20 ms until we began sending 301 bytes at 500 Hz (a much higher rate than we will be transmitting in the JADS federation). There were no lost messages in the tests below 301 bytes at 500 Hz. In the tests that lost data, the latencies became over 100 ms (up to 450 ms).

After characterizing the network and the RTI in the simple one-way tests, JADS wanted to determine whether the RTI would support the anticipated loads placed on it by the JADS federation. A test federate was needed that could simulate these kinds of loads. A *testfed* federate was developed to satisfy these requirements. The *testfed* federate accepts command line arguments that specify the characteristics of an instance of the federate. The user can specify the federate ID number (-f), the duration of the test (-d), the size of the attributes and interactions (-s), the rate that attributes are published (-r), the number of updates at the specified rate (-n), the amount of time the federate should wait before starting to publish at its specified rate (-w), and whether interactions should be published (-i). There is an additional argument (-c) that indicates which federate is the controller. There must be one and only one controller federate in the *testfed* federation. There is only one attribute and one interaction used by all federates. All federates subscribe to the attribute and the interaction.

After extensive testing, we provided our RTI 1.3 results to DMSO along with the information we learned (see Lessons Learned below). The RTI 1.3 development team modified the RTI and incorporated other modifications into RTI 1.3-2 with the objective of improving performance for reliable traffic. In the RTI 1.3-2 reliable tests, the performance of reliable traffic drastically improved. With the sender publishing up to 301 bytes at 400 Hz, the minimum latencies were between 8 and 12 ms, and the maximum latencies were between 10 and 200 ms. JADS testers observed intermittent latency spikes that caused problems. When a federate tried to publish 301 bytes at 400 Hz, reliable data was lost. When it tried to publish 501 bytes at 400 Hz, a federate would crash. These problems never occurred in previous tests of RTI versions. Further verification of these results have not been made by JADS using the most current RTI version.

Tests using a three node communications architecture were designed to assist JADS in optimizing the performance of the RTI as well as the JADS EW phase 2 test federation components. The expanded test environment used at least 3 and as many as 6 SGI O2 workstations (either R5000 or R10000 models running IRIX 6.3) representing the actual test configuration. JADS ran realistic tests with six federates on six computers on three network nodes. The six-federate tests produced a wide variety of results. We had a few runs where only one or two best effort attributes were lost and the maximum latency was less than 50 ms. There were some runs that had up to 100 attributes lost and an occasional high interaction latency of between 1 and 8 seconds and there were some runs that had federates that crashed. We reported these results to DMSO. Subsequently, the DMSO RTI development team found a software "bug" that limited the number of federates that could execute in a federation.

## Lessons Learned

JADS experience with testing versions of the RTI proved to be an iterative process of identifying problems and implementing solutions. The problems related to the RTI were discussed directly with DMSO who worked with JADS to resolve the problems.

1. Time-To-Live - In the initial tests we performed with RTI 1.0-2, best effort traffic was not received at any computer on a different LAN. Using a network sniffer to look at the network data packets, JADS discovered that the Time-To-Live value was set to 1. A packet's Time-To-Live value indicates how many hops it can take before it is discarded by the network. A value of 1 does not allow a packet to exit the LAN. A federation running with RTI 1.0-2 out of the box would not allow federates to communicate best effort traffic outside of a LAN. Using the JADS 2 node network configuration (shown in Figure 5) required network data packets to cross from one LAN through the routers (Micro-IDNX-20) to reach the test federate on another LAN mirroring the JADS EW phase 2 network architecture. JADS was provided a special library from DMSO that allowed us to use RTI 1.0-2 across our network communications gear. Subsequent versions of the RTI provide for a user-defined parameter value in the RID file to set Time-To-Live.

2. TCP No Delay and the Nagle Algorithm - Prior to RTI version 1.3-2, the RTI runs with a default setting for the TCP_NODELAY socket option. On the SGIs, the default value is FALSE. This means that the Nagle algorithm [3] will be in effect for both attribute and interaction data sent reliable. If data is published using reliable transportation at data rates at or above 5 Hz, then the latency of the data is increased significantly. As a result of sharing this information with RTI developers, RTI version 1.3-2 sets the TCP_NODELAY option to TRUE, disabling the Nagle algorithm.

3. Tick - As we implemented and experimented with tick during initial test runs with each RTI release, we learned how important it is to understand how tick works in its various forms in order to tune a federation properly. The tick function is how a federate transfers process control to the RTI so it can do its work. The user's federates must constantly tick the RTI or nothing will happen in the federation. There are two variations to tick: one has no parameters (tick ( )), the other has a minimum and maximum value (tick (min, max)). The tick function called with no parameters empties its queue before it returns to the federate which could starve the federate from getting its necessary processor time.

The tick function called with a minimum and maximum value (tick (min, max)) will stay in tick at least the amount of time specified by the minimum parameter, but no longer than the maximum parameter. If the RTI empties its queue before the minimum time elapses, it will try to sleep for the rest of the time. On an SGI, this is a problem because the minimum sleep (i.e., sginap or select) time is 10ms. As a result, if the RTI user specifies a minimum value of 10 ms and the RTI uses 9 ms to do its work, an SGI will sleep for an additional 10 ms. If zero or some small number is specified for the minimum, the RTI will not sleep. This can cause the federate/RTI to use as much as 90% of the CPU. We benefited greatly from open communication with DMSO about features of tick and verifying the results we obtained from different settings. Unfortunately, we did not find any documentation source for tick features and tuning ideas. We advised DMSO that this information is very beneficial to all but the casual RTI user. Each federation and its architecture is different and it will require some experimentation by the federation developers to find the optimum use of tick.

4. Initial Publication Rates - When a federate starts, we found that it is best if it publishes some initial data at low data rates to set up the network. In the JADS tests with three federates (one published 11 updates at 20 Hz, one published 2 updates at 20 Hz, and one published one update at 20 Hz), best effort data was lost and reliable data had high latencies in the initial burst of data. When we added a 5 second delay at the start, during which the federates published data at 1 Hz, these startup problems were eliminated.

5. Fast malloc - There is a library on SGIs that provides a faster version of *malloc* (used to dynamically allocate memory). To use this library it must be linked with user software with the *-lmalloc* option. In an attempt to make it as efficient as possible, the JADS RTI logger was linked with this library. While running RTI tests linked with the logger, the federate would crash after it resigned from the federation. After speaking with the DMSO team, they said they were aware of problems using this library and recommended not using it.

6. Network Optimization - During a conversation to discuss the latency problems, representatives at ACETEF mentioned that their facility used Ethernet switches to eliminate problems that might be caused by collisions over a shared Ethernet configuration like the JADS test bed. JADS purchased and installed an 8-port Ethernet switch for use in the dedicated test bed. This increased the test bed LAN's performance well beyond that of the previous 10 Mbps, half duplex LAN. Recent two-federate testing with the *testfed* tool for a 5-minute test has shown that the Ethernet collision rate has dropped from about 500 - 1000 collisions to 0 collisions! While this LAN improvement has not eliminated all the latency problems observed during tests of RTI 1.3-2 (because Ethernet collisions were not the only cause of them), it seems to have significantly reduced, directly or indirectly, the frequency of the 1-second class latency events.

7. All TCP Implementations Are Not the Same - The RTI is developed on Sun/Solaris machines. It has been optimized to run in this environment. The implementation of TCP on SGI/IRIX machines (and others) is different than the implementation of TCP on Sun/Solaris. Since TCP is a major Internet protocol, one might naively assume all TCP implementations were developed and tested to conform to the same "standards" and they all operate in more-or-less the same manner. JADS found information in the TCP literature[4] showing this naive view to be incorrect. For example, the current IRIX TCP on the SGI systems at JADS, AFEWES, and ACETEF is very likely to be one of the "Berkeley-derived implementations", but the current Solaris TCP used by the RTI's developer is very likely to be an

"independent" implementation by Sun Microsystems. Furthermore, various TCP implementations are known to respond differently in time-critical situations. This suggests that distributed simulations running the same federates over the same LANs and WANs might exhibit different behavior on SGIs than on Sun/Solaris machines because of differences in the underlying TCP implementations.

## Summary

This paper documents the JADS EW test methodology, accumulated experience in implementing an HLA test architecture, and the current results of JADS RTI test activities conducted. Based upon the latency values measured for the most recent RTI software release, further tests will be conducted as further resolution of latency problems are accomplished by DMSO and JADS. As documented in this paper, much has been accomplished and learned by both JADS and DMSO's RTI team from this effort. The progress made and lessons learned so far represent a significant advance in characterizing and reducing RTI latency as well as improving RTI features for T&E federations. DMSO has provided significant support to address RTI problems as they were discovered.

## References

1. D. Wright and C. Harris: "Testing RTI/Network Interactions for Latency", SISO Spring 98 Simulation Interoperability Workshop, paper 98S-SIW-152.

2 D. Wright and C. Harris: "Determining and Expressing RTI Requirements", SISO Spring 98 Simulation Interoperability Workshop, paper 98S-SIW-158.

3 W. Richard Stevens: "TCP Interactive Data Flow" TCP/IP Illustrated - Volume 1, Addison-Wesley, Massachusetts 1994.

4 Vern Paxson: "Automated Packet Trace Analysis of TCP Implementations", Proceedings of SIGCOMM '97, technical paper